IN THE CLAIMS

Please cancel claims 26-30 without prejudice or disclaimer.

Applicants note that claim 25 is amended not to overcome prior art but to correct a typographical error. The amendment to claim 25 is not narrowing scope and therefore no prosecution history estoppel arises from the amendment to claim 25. See *Festo Corp. v. Shoketsu Kinzoku Kogyo Kabushiki Co.*, 56 USPQ2d 1865, 1870 (Fed. Cir. 2000).

Please amend the following claim:

Please amend Claim 25 as follows:

A1

25. (Amended) The computer program product as recited in claim 24, wherein the recovery process further comprises the program step of:

if less than a specified number of the nodes voted to continue, performing the recovery process on the specified number of the nodes.

REMARKS

Claims 1-30 were pending in the Application. Claims 1-30 were rejected under under 35 U.S.C. §103(a). Claims 26 -30 are cancelled without prejudice or disclaimer. Applicants respectfully traverse the rejections and respectfully request that the Examiner reconsider and withdraw all outstanding rejections.

Attached hereto is a marked-up version of the changes made to the claims by the current reply. The attached page is captioned "**Version with markings to show changes made.**"

I. REJECTIONS UNDER 35 U.S.C. §103(a):

The Office Action has rejected claims 1-30 as being unpatentable over San Andres et al. (U.S. Patent No. 5,956,489) (hereinafter "San Andres"). Applicants respectfully traverse the rejections and respectfully request that the Examiner reconsider and withdraw all outstanding rejections.

A *prima facie* showing of obviousness requires the Examiner to establish, *inter alia*, that the prior art references teach or suggest, either alone or in combination, all of the limitations of the claimed invention, and the Examiner must provide a motivation or suggestion to combine or modify the prior art reference to make the claimed inventions. See M.P.E.P.§2142. The motivation or suggestion to combine references must come from one of three possible sources: the nature of the problem to be solved, the teachings of the prior art, and the knowledge of persons of ordinary skill in the art. See *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998). The showings must be clear and particular. See *In re Dembiczak*, 50 U.S.P.Q.2d 1614, 1617 (Fed. Cir. 1999). Broad conclusory statements regarding the teaching of multiple references, standing alone, are not evidence. *Id.*

In order to reject under 35 U.S.C.§103, therefore, the Examiner must provide a proper motivation for combining or modifying the references. See M.P.E.P.§2142; *In*

re Rouffet, 47 U.S.P.Q.2d 1453, 1457-1458 (Fed. Cir. 1998). The Examiner recites that "it would have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the teaching of San Andres with the step of each node in the computer cluster voting *based on a functional outcome of the database update request.* This *modification would allow the teachings of San Andres to provide access to identical data and so that the on line service appears the same to all end users.*" See Office Action, Pages 3-4.

There is no motivation to modify San Andres as the *proposed modification would render the invention in San Andres unsatisfactory for its intended purpose* and therefore there is no suggestion or motivation to make the proposed modification. See *In re Gordon*, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984); M.P.E.P. §2143.01. Furthermore, the *proposed modification would change the principle of operation of San Andres* and therefore the teachings of San Andres are not sufficient to render the claims *prima facie* obvious. See *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959); M.P.E.P. §2143.01. San Andres teaches that "the servers 120 that receive the update transaction from the Arbiter respond by processing the update transaction, and by returning a status code that indicates the success or failure of the transaction." See Column 19, Lines 25-28. San Andres further teaches that "the update transactions are *dispatched to the servers 120 in a serialized order ... so that all servers of the service group process the update transactions in the same order. This ensures consistency between the replicated copies of the service content data.* Each time an update transaction is dispatched by the Arbiter, the Arbiter monitors the outcome ("success" or "failure") of the transaction on each server 120 by checking the status codes returned by the servers 120. *When one server 120 of the service group processes the dispatched*

transaction differently than the other servers, the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group. In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent" with the final outcome, and are taken off-line." See Column 19, Lines 37-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off line for maintenance.*

San Andres further teaches that "what is also needed is an efficient mechanism for bringing the content of an application server up-to-date with that of other application servers ... *so that existing application servers can efficiently be taken off-line for maintenance.*" See Column 2, Lines 23-30. As interpreted by the Applicants, *the purpose of San Andres is to ensure that existing application servers can efficiently be taken off-line for maintenance. However, by having each application server implement a voting scheme to indicate whether there is a difference between the broadcast results and the local modification-result code, the Arbiter would not be able to decide which application servers are inconsistent and thus be taken off-line for maintenance.* Hence, the *proposed modification would render the invention in San Andres unsatisfactory for its intended purpose* and therefore there is no suggestion or motivation to make the proposed modification. See *In re Gordon*, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984); M.P.E.P. §2143.01. Furthermore, the *proposed modification would change the principle of operation of San Andres* and therefore the teachings of San Andres are not

sufficient to render the claims *prima facie* obvious as a matter of law. See *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959); M.P.E.P. §2143.01.

San Andres does not teach or suggest "*each node* in the computer cluster *voting based on a functional outcome of the database update request*" as recited in claim 1. San Andres does not teach or suggest "*voting, by all of the other nodes in the computer cluster*, to approve update if a match results from the comparison" as recited in claims 9 and 21 and similarly claim 15. The Office Action states that "San Andres does not explicitly indicate the step of each node in the computer cluster voting based on a functional outcome of the database update request." See Office Action, Page 3. The Office Action further states that "it would have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the teaching of San Andres with the step of each node in the computer cluster voting *based on a functional outcome of the database update request*. This *modification would allow the teachings of San Andres to provide access to identical data and so that the on line service appears the same to all end users*. " See Office Action, Pages 3-4. For at least the reasons stated above, Applicants assert that it would not be obvious to modify the teachings of San Andres with the step of each node in the computer cluster voting *based on a functional outcome of the database update request* and therefore the Examiner has not presented a *prima facie* case of obviousness. Furthermore, for at least the reasons stated above, Applicants assert that it would not be obvious to modify the teachings of San Andres with the step of *voting by all of the other nodes in the computer cluster* and therefore the Examiner has not presented a *prima facie* case of obviousness.

San Andres does not teach or suggest "*detecting* an *out-of-sync condition* as a *result of a different functional outcome*" as recited in claim 1. Instead, San Andres teaches that "*when one server 120 of the service group processes the dispatched transaction differently than the other servers, the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group*. In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance.* As interpreted by the Applicants, *San Andres simply teaches an Arbiter using a scheme to determine which servers are to be taken off-line but does not teach detecting an out-of-sync condition as a result of a different functional outcome.*

San Andres does not teach or suggest "*applying the update to a local copy of the database* at each of the plurality of nodes in the computer cluster" as recited in claims 9 and 21 and similarly in claim 15. Instead, San Andres teaches that "the servers 120 that receive the update transaction from the Arbiter *respond by processing the update transaction*, and by *returning a status code that indicates the success or failure of the transaction.*" See Column 19, Lines 25-28. San Andres further teaches that "*all replicated servers of the service group maintain local copies of the service's content data*, and provide user access to such data." See Column 9, Lines 19-21. As interpreted by the Applicants, San Andres teaches that the replicated servers maintain local copies

of the service's content data. Furthermore, as interpreted by the Applicants, San Andres teaches that the servers process the update transaction received from the Arbiter. However, as interpreted by the Applicants, San Andres *does not teach that the servers apply the update to a local copy of the database.* San Andres simply teaches that the servers process the update transaction. Applicants kindly request the Examiner to particularly point out in San Andres where the servers *apply the update to a local copy of the database.*

San Andres does not teach or suggest that the "*node requesting update broadcasts results of update to all of the other nodes in the computer cluster*" as recited in claims 9 and 21 and similarly in claim 15. Instead, San Andres teaches that "when an application server of a service group receives a client request that indicates a modification to replicated service content data, the *server/service generates an update transaction and sends the update transaction to the Arbiter.* The Arbiter records the update transaction in a service-group specific transaction log ... and forwards the transaction for immediate processing to every application server in the server group ... The *application servers process the update transaction, and return status codes indicating, for each respective application server, the 'success' or 'failure' of the transaction.*" See Column 3, Lines 26-38. As interpreted by the Applicants, San Andres teaches that the server receiving the request that indicates a modification generates an update transaction and sends the update transaction to the Arbiter. As interpreted by the Applicants, San Andres further teaches that the Arbiter forwards the update transaction to each server and that each server *returns a status code to the Arbiter* indicating the success or failure of the transaction. San Andres does not teach that the node requesting

an update *broadcasts the results of the update to all of the other nodes* in the computer cluster.

San Andres does not teach or suggest "*comparing, by all of the other nodes* in the computer cluster, the *update results to results of application of the update to the local copy of the database*" as recited in claims 9 and 21 and similarly in claim 15. Instead, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group*. In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance. San Andres does not teach that *all of the other nodes in the computer cluster compare* the update results to results of application of the update to the local copy of the database. Furthermore, San Andres does not teach that all of the other nodes in the computer cluster compare the *update results to results of application of the update to the local copy of the database.*

San Andres does not teach or suggest "voting, by all of the other nodes in the computer cluster, *to approve update if a match results from the comparison*" as recited in claims 9 and 21 and similarly in claim 15. Applicants kindly request the Examiner to

particularly point out in San Andres where all of the other nodes in the computer cluster vote to *approve update if a match results from the comparison.*

San Andres does not teach or suggest "a *group services client* operable for *broadcasting an update to a database* shared among a plurality of nodes in the computer cluster" as recited in claim 9. Instead, San Andres teaches a "host data center 104" that "comprises a *plurality of application servers (APP servers) 120 connected to a high speed local area network (LAN)* 122." See Column 5, Lines 18-20. Furthermore, San Andres teaches a "*host data center 104*" that "*includes multiple Arbiter microcomputers 128* that run the Arbiter service application." See Column 6, Lines 33-34. Furthermore, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group.* In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off line for maintenance. As interpreted by the Applicants, the *Arbiter* is an arbitration mechanism and *is not a group services client.*

San Andres does not teach or suggest "*voting, by any one of the other nodes* in the computer cluster, to *continue with update process if a match does not result from the comparison*" as recited in claim 21. Instead, San Andres teaches that "when one server

120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group*. In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance*. San Andres does not teach *voting by any of the other nodes in the computer cluster*. Furthermore, San Andres does not teach voting by any of the other nodes in the computer cluster *to continue with update process if a match does not result from the comparison*

For at least the above reasons, claims 1, 9, 15 and 21 are patentable over San Andres. Claims 2-8, 11-14, 16-20 and 22-25 each recite combinations of features including the above combinations, and thus are patentable for at least the above reasons as well. Claims 2-8, 11-14, 16-20 and 22-25 recite additional features which, in combination with the features of the claims upon which they depend, are patentable over San Andres.

For example, San Andres does not teach or suggest that "the *out-of-sync condition is an error*" as recited in claim 2. Instead, San Andres teaches that "*when one server 120 of the service group processes the dispatched transaction differently than the other servers, the Arbiter uses a voting scheme to decide which server or servers are to be*

taken off-line within the service group." In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance.* As interpreted by the Applicants, *San Andres simply teaches an Arbiter using a scheme to determine which servers are to be taken off-line but does not teach detecting an out-of-sync condition that is an error.*

San Andres does not teach or suggest that "*refreshing* the database *in response to the detecting step*" as recited in claim 3. Instead, San Andres teaches that "when an application server of a service group receives a client request that indicates a modification to replicated service content data, the server/service generates an update transaction and sends the update transaction to the Arbiter. The Arbiter records the update transaction in a service-group specific transaction log ... and *forwards the transaction for immediate processing to every application server* in the server group ... *The application servers process the update transaction, and return status codes indicating, for each respective application server, the 'success' or 'failure' of the transaction.*" See Column 3, Lines 26-38. San Andres further teaches that "*when different application servers return different status codes, the Arbiter uses the ... conflict resolution feature to resolve the conflict.*" See Column 3, Lines 41-44. As interpreted by the Applicants, San Andres teaches an arbiter that sends an update transaction to each application server to be processed. Upon the application servers processing the update

transactions, the arbiter may be configured to resolve a conflict where the conflict being different status codes sent by different application servers upon processing the update transaction. As interpreted by the Applicants, the *application servers do not process the update transaction in response to detecting an out-of-sync condition.* The *application servers simply process the update transaction sent by the arbiter.*

San Andres does not teach or suggest that "*resetting cluster membership in response to the detecting step*" as recited in claim 4. Instead, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group.* In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance.* As interpreted by the Applicants, *San Andres simply teaches an Arbiter using a scheme to determine which servers are to be taken off-line but does not teach resetting cluster membership in response to detecting an out-of-sync condition.*

San Andres does not teach or suggest "declaring an end-of-transaction state on *update voting completion* when the database update is being done in a transactional manner" as recited in claim 6. Instead, San Andres teaches that "the servers that receive

the update transaction from the Arbiter respond by processing the update transaction, and by returning a status code that *indicates the success or failure of the transaction*." See Column 19, Lines 25-28. As interpreted by the Applicants, San Andres teaches that the servers process the update transaction sent from the Arbiter and then return a status code that indicates whether the *server was successful or not on processing the update transaction*. San Andres does not teach that the servers declare an end-of-transaction state *on update voting completion* when the database update is being done in a transactional manner.

San Andres does not teach or suggest "*backing out an update when update voting does not meet a criteria established for success*" as recited in claim 7. Instead, San Andres teaches that "the servers that receive the update transaction from the Arbiter respond by processing the update transaction, and by returning a status code that *indicates the success or failure of the transaction*." See Column 19, Lines 25-28. As interpreted by the Applicants, San Andres simply teaches that the servers process the update transaction sent from the Arbiter and then return a status code that indicates whether the *server was successful or not on processing the update transaction*. San Andres does not teach that the Arbiter or server application *backs out an update when the update voting does not meet a criteria established for success*.

San Andres does not teach or suggest that "the criteria for success is that *no more than one node has inconsistent results*" as recited in claim 8. Instead, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group*. In the general

case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance.* San Andres *does not teach that the Arbiter's voting criteria is that no more than one node has inconsistent results.* Instead, as interpreted by the Applicants, the Arbiter's voting criteria is to treat the *minority number* of servers *with a different outcome than the majority number* of servers *as inconsistent.*

San Andres does not teach or suggest "*voting, by any one of the other nodes* in the computer cluster, to *continue with update process if a match does not result from the comparison*" as recited in claims 10 and 26 and similarly in claim 16. Instead, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group.* In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance.* San Andres does not

teach that *voting by any of the other nodes in the computer cluster*. Furthermore, San Andres does not teach voting by any of the other nodes in the computer cluster *to continue with update process if a match does not result from the comparison*.

San Andres does not teach or suggest "*broadcasting an approval* of the *update to the database if all of the other nodes vote to approve the update*" as recited in claims 11, 22 and 27 and similarly in claim 17. Instead, San Andres teaches that "each update transaction replicated by the transaction replication service is stored in a transaction log. When a new application server is brought on-line, *previously-dispatched update transactions stored in the transaction log are dispatched in sequence to the new server* to bring the new server's content data up-to-date." See Abstract. San Andres does not teach or suggest *broadcasting an approval* of the *update to the database*. Furthermore, San Andres does not teach or suggest broadcasting an approval of the update to the database *if all of the other nodes vote to approve the update*. Applicants kindly request the Examiner to particularly point out in San Andres where *broadcasting an approval* of the *update to the database* and broadcasting an approval of the update to the database *if all of the other nodes vote to approve the update* is taught.

San Andres does not teach or suggest "*if more than one of the plurality of nodes votes to continue,* performing a recovery process" as recited in claims 12, 23 and 28 and similarly in claim 18. Instead, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group*. In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the*

service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent" with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance.* As stated above, San Andres *does not teach voting by nodes.* Furthermore, San Andres does not teach *if more than one of the plurality of nodes votes to continue*, performing a recovery process.

San Andres does not teach or suggest "*if more than a specified number of the nodes vote to continue, backing out the update to the database* as recited in claim 13, 24 and 29 and similarly in claim 19. Instead, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group.* In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance.* As stated above, San Andres *does not teach voting by nodes.* Furthermore, San Andres does not teach *if more than a specified number of the nodes vote to continue backing out the update to the database.*

San Andres does not teach or suggest "*if less than a specified number of the nodes voted to continue, performing the recovery process on the specified number of the nodes*" as recited in claims 14, 25 and 30 and similarly in claim 20. Instead, San Andres teaches that "when one server 120 of the service group processes the dispatched transaction differently than the other servers, *the Arbiter uses a voting scheme to decide which server or servers are to be taken off-line within the service group*. In the general case, the Arbiter uses a "majority rules" voting scheme. Under the majority rules scheme, *if a minority number of servers 120 of the service group report a different outcome than the other servers, the minority servers are treated as being "inconsistent"* with the final outcome, and are taken off-line." See Column 19, Lines 46-55. As interpreted by the Applicants, San Andres teaches that an *Arbiter* may *use a voting scheme to decide which application servers are deemed to be consistent and take application servers that are inconsistent off-line for maintenance*. As stated above, San Andres *does not teach voting by nodes*. Furthermore, San Andres does not teach *if less than a specified number of the nodes voted to continue, performing the recovery process on the specified number of the nodes*.

As a result of the foregoing, Applicants respectfully assert that the Examiner's *prima facie* case of obviousness is not taught or suggested by the cited prior art since there are numerous claim limitations, and thus one skilled in the art would not have been able to create the claimed invention in view of the cited prior art.

II. CONCLUSION

As a result of the foregoing, it is asserted by Applicants that the remaining Claims in the Application are in condition for allowance, and respectfully request an early allowance of such Claims.

Applicants respectfully request that the Examiner call Applicants' attorney at the below listed number if the Examiner believes that such a discussion would be helpful in resolving any remaining problems.

Respectfully submitted,
WINSTEAD SECHREST & MINICK P.C.
Attorneys for Applicants

By: _____
Robert A. Voigt, Jr.
Reg. No. 47,159
Kelly K. Kordzik
Reg. No. 36,571

5400 Renaissance Tower
1201 Elm Street
Dallas, Texas 75270-2199
(512) 370-2832

::ODMA\PCDOCS\AUSTIN_1\169249\1
1162:7047-P280US

VERSION WITH MARKINGS TO SHOW CHANGES MADE

In the claims:

Please amend Claim 25 as follows:

25. (Amended) The computer program product as recited in claim 24, wherein the recovery process further comprises the program step of:

if less than a specified number of the nodes voted to continue, performing the recovery process on the specified number of the nodes.

[A method for maintaining a consistent set of replicas of a database within a computer cluster, comprising the steps of:

broadcasting an update to a database shared among a plurality of nodes in the computer cluster;

applying the update to a local copy of the database at each of the plurality of nodes in the computer cluster;

node requesting update broadcasts results of update to all of the other nodes in the computer cluster;

comparing, by all of the other nodes in the computer cluster, the update results to results of application of the update to the local copy of the database; and

voting, by all of the other nodes in the computer cluster, to approve update if a match results from the comparison.]